Project 1: Battleship

Retrospective Write-Up

Nolan Blankenau, Dawson Frick, Aaron Gearhart, Harry Heeb, Nick North

Meeting Notes

Meeting 1: 9/6/2019

Location: Spahr Classroom

Attendance: Everyone

* NodeJS
* Paperjs for graphics
* Use javascript sound libraries
* Decided to meet tomorrow to decide on work division and features
* <https://codeburst.io/getting-started-with-node-js-a-beginners-guide-b03e25bca71b>

Meeting 2: 9/7/2019

Location: Google Hangouts

Attendance: Everyone

* Features
  + OnClick Events for selecting coordinate
    - Dragging feature for selecting coordinates (setup)
    - Rotate ships
  + Sounds
    - Hit
    - Miss
    - Destroyed ship
  + Colored Graphics on grid
    - Red - hit
    - Green - miss
    - Grey - Setup
  + Objects
    - Board
      * Squares
      * Grey out board after player switch, countdown
    - Boats
    - Player Object
      * Select how many boats are spawned
    - Executive
      * Graphics
      * Pulling everything together
* Work divide:
  + 2 people working on Graphics/Sound implementation
    - Nolan, Dawson
      * Melonjs
  + 3 people working on Objects/Functionality
    - Aaron, Harry, Nick
* Made notes on setup of each object, will discuss “header file” setup at a later date

Meeting 3: 9/9/2019

Location: Eaton Classroom

Attendance: Everyone

* Scrum
  + Saturday’s meeting was a collaborative effort, everyone gave input on design, features, and work division.
  + Next meeting (Lab today 9/9/2019) we will divide into our tasks and start preliminary work on the code like setting up what functions we will use. Also we will setup the GitHub repository for the project.

Meeting 4: 9/9/2019

Location: Eaton 1005D & Spahr 2320 (LAB)

Attendance: Everyone

[Repo](https://github.com/apgearhart1/EECS448-Project1)

* **Switched language to Python3**
  + Using pygame library
* Discussed that we should start with getting the executive class and board class working first.
* Started on functionality of boat class

Meeting 5: 9/12/2019

Location: Google Hangouts

Attendance: Everyone

* Everyone gave an update of what they worked on
* Discussed documentation standards in within the files
* Discussed standard for passing in coordinates between files
  + 2D Array

Meeting 6: 9/13/2019

Location: Spahr Classroom

Attendance: Everyone

* Discussed what everyone did the previous night since some people left the Google Hangouts earlier than others
* Decided to use Sphinx as our documentation tool
* Discussed state screens needed - all in one file
  + Welcome screen
  + Boat selection screen
  + Boat placement screen
  + Game play screen
  + Player transition screen

Meeting 7: 9/16/2019

Location: Spahr Classroom & Spahr 1322

Attendance: Everyone

* Discussed what we did last night
  + Changes to board.py
* Started executive.py
* Began work on state-screens

Meeting 8: 9/17/2019

Location: Google Hangouts

Attendance: Everyone

* Discussed what we did last
  + Made the welcome screen interactive and lets players select boats
  + Still some bugs that need to be fixed
* Adding UI functionality for switching players

Meeting 9: 9/18/2019

Location: Spahr Classroom

Attendance: Everyone

* Discussed what we did last night
  + Player switching works
  + Player can now select number of boats
* Things left to do
  + Win screen
  + Player switch screen (countdown)
  + Sunken boat indicator
  + Graphics for coordinates, letters for columns and numbers for rows
  + Documentation/Citations

Meeting 10: 9/21/2019

Location: Spahr 1324

Attendance: Aaron, Harry, Dawson, Nolan

* Time to finish the project!
  + Documentation is done

Retrospective

1. Description on how work was split between teammates

When we originally split up work we had Dawson and Nolan working on the front-end. On the back-end was Aaron, Harry, and Nick. This structure was followed for some time. Dawson and Nolan worked within main.py programming the basic graphics for Battleboats. Harry wrote a board class, Aaron wrote a class for boats, and a player class was made by Nick. After this, Aaron and Harry started developing in main.py to implement gameplay logic. Nick went on to write our executive class. Toward the end, Nick was implementing the executive class into main.py while everyone else finished adding graphics and game logic.

1. Challenges and how they were overcome or dealt with

Some challenges we faced as a team was dividing up work and then making sure what we were working on functioned correctly in the end. Initially, we divided up the work by giving each of us an element or class of the project. We had a player class, boat class, board class, executive class and main file as well. The main is where we would run most of our graphics and user interactions on. When it came down to bring it all together, sometimes we would have trouble figuring out what each of our functions did, that’s where good documentation came in. Also, we had to keep good communication through Slack because we were all working on the project at different times when we weren’t actually in a meeting and sometimes that would lead to us writing redundant code which can misuse time. Overall, we were able to diminish challenges that we came across by taking a second and thinking about what we were going to do and communicating that to the group so everyone was on the same page.

1. Any features that did not make the demo version

There were a few features that were discussed, but never made it into the demo version of our game. We wanted to include sound bites for hits and misses as well as background music to play throughout the duration of the game, but we didn’t get to it. There was also an idea early on to add a mode where you could play a computer, but we soon realized this wouldn’t be reasonable to accomplish before the deadline.

1. Retrospective on what the team would have done different

One of the most important things we could’ve done differently is how we planned the project at meetings. If we had spent more time talking about which classes handled which features we would’ve been able to cut down on redundant code. We could’ve also spent more time discussing the parameters of our functions because it isn’t always clear what types certain variables are in Python. Another thing we could’ve done differently is to make better use of all of the code we write. Some of the code we’ve written for each of the classes was never actually used in the final demo because it wasn’t necessary. Lastly, we could’ve communicated more frequently to set deadlines for certain parts of the project so that we don’t procrastinate.